BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

LINES

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

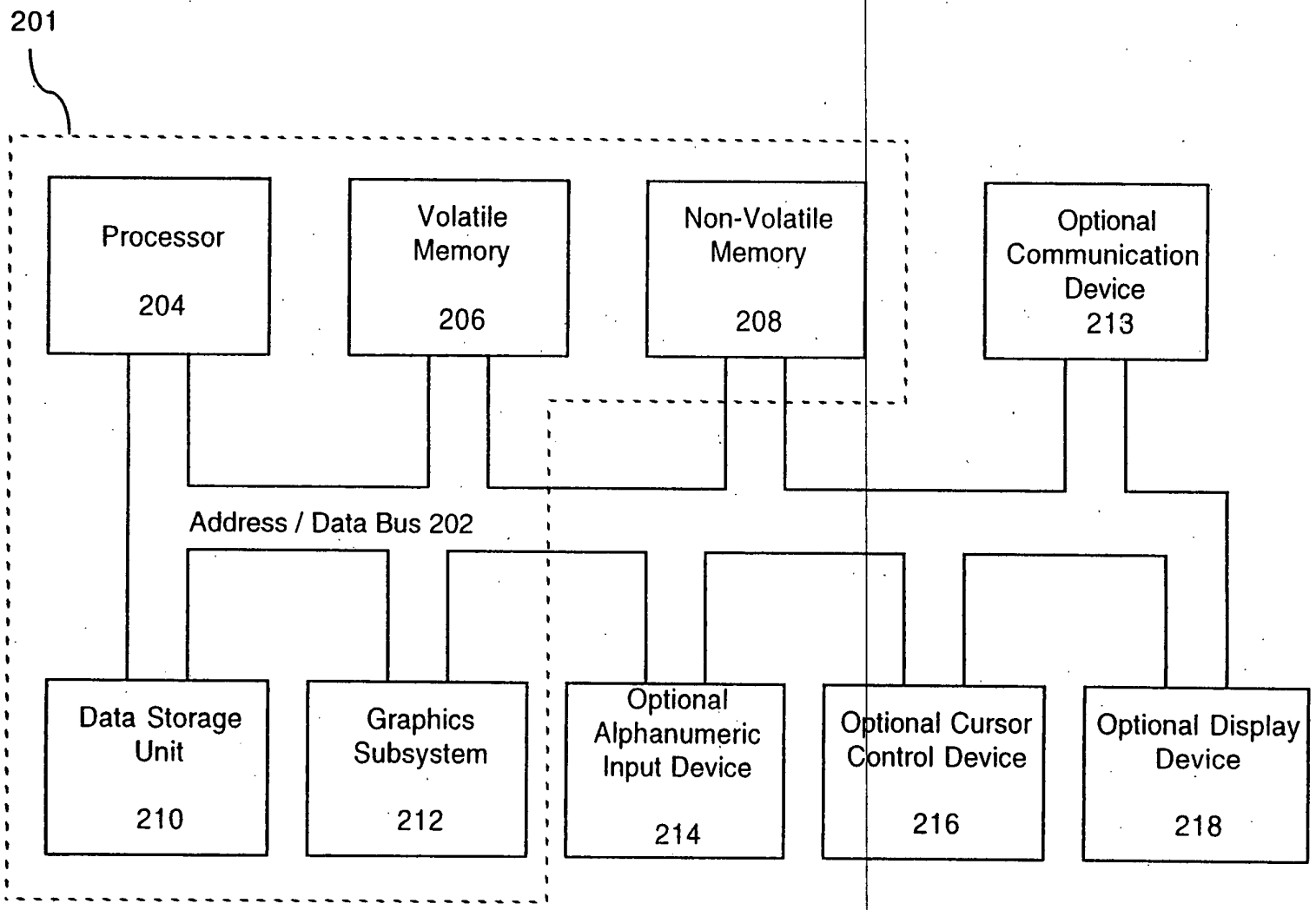
201
Processor
204
Volatile
Memory
206
Non-Volatile
Memory
208
Optional
Communication
Device
213
Address / Data Bus 202
Data Storage
Unit
210
Graphics
Subsystem
212
Optional
Alphanumeric
Input Device
214
Optional Cursor
Control Device
216
Optional Display
Device
218

Figure 1

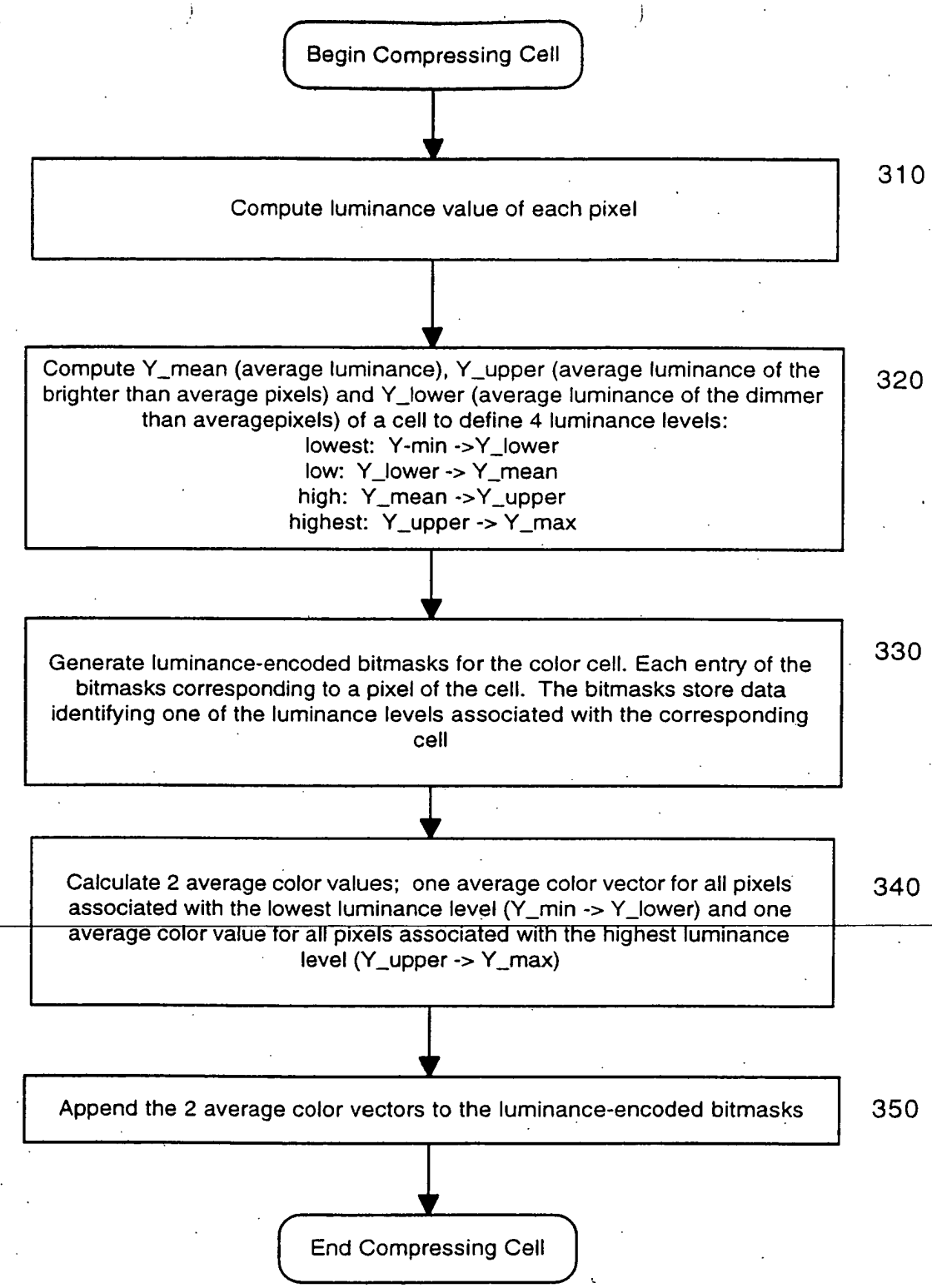
300
Begin Compressing Cell
Compute luminance value of each pixel
310
Compute Y_mean (average luminance), Y_upper (average luminance of the brighter than average pixels) and Y_lower (average luminance of the dimmer than averagepixels) of a cell to define 4 luminance levels:
lowest: Y-min ->Y_lower
low: Y_lower -> Y_mean
high: Y_mean ->Y_upper
highest: Y_upper -> Y_max
320
Generate luminance-encoded bitmasks for the color cell. Each entry of the bitmasks corresponding to a pixel of the cell. The bitmasks store data identifying one of the luminance levels associated with the corresponding cell
330
Calculate 2 average color values; one average color vector for all pixels associated with the lowest luminance level (Y_min -> Y_lower) and one average color value for all pixels associated with the highest luminance level (Y_upper -> Y_max)
340
Append the 2 average color vectors to the luminance-encoded bitmasks
350
End Compressing Cell

FIG. 2

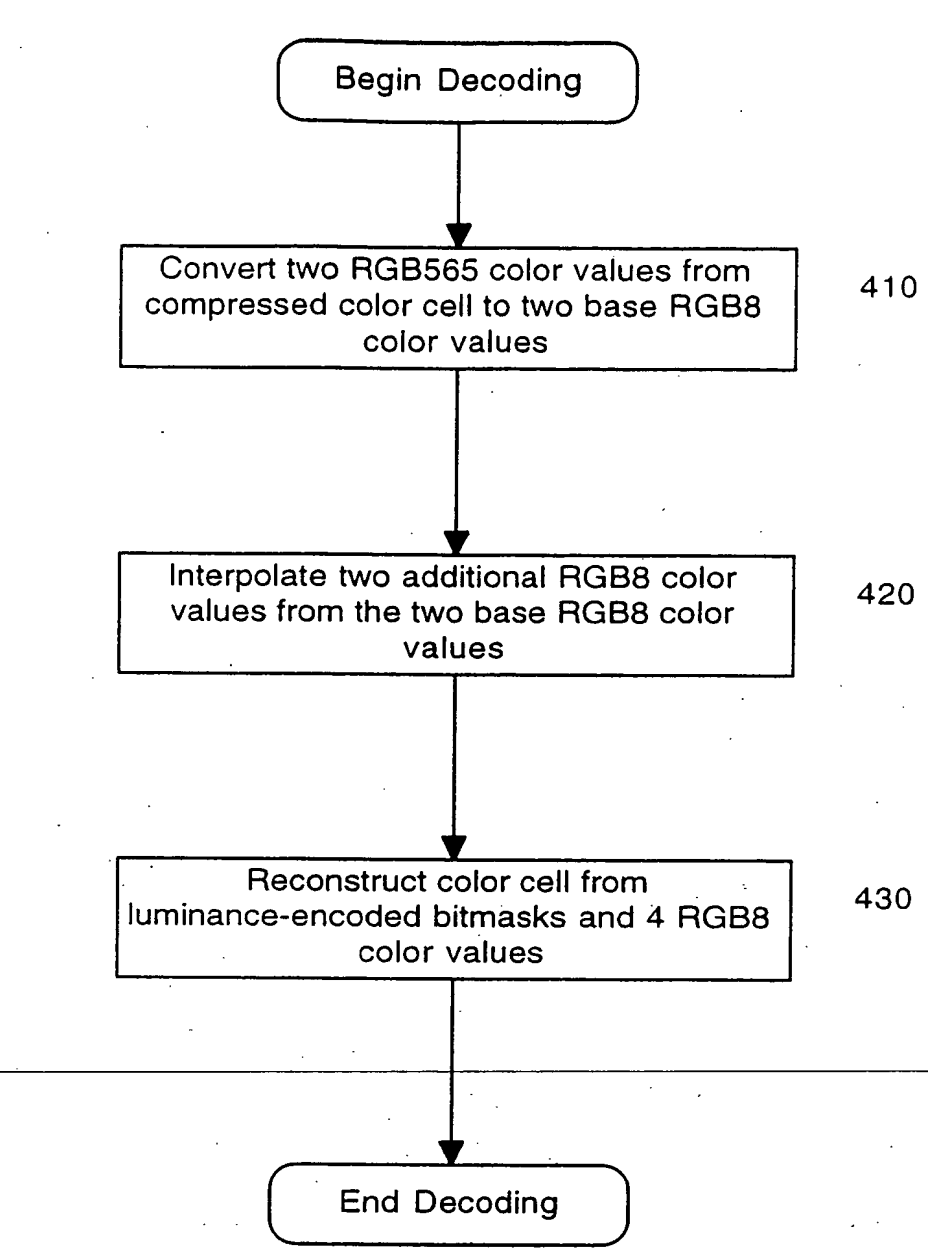
400
Begin Decoding
Convert two RGB565 color values from compressed color cell to two base RGB8 color values
410
Interpolate two additional RGB8 color values from the two base RGB8 color values
420
Reconstruct color cell from luminance-encoded bitmasks and 4 RGB8 color values
430
End Decoding

FIG. 3

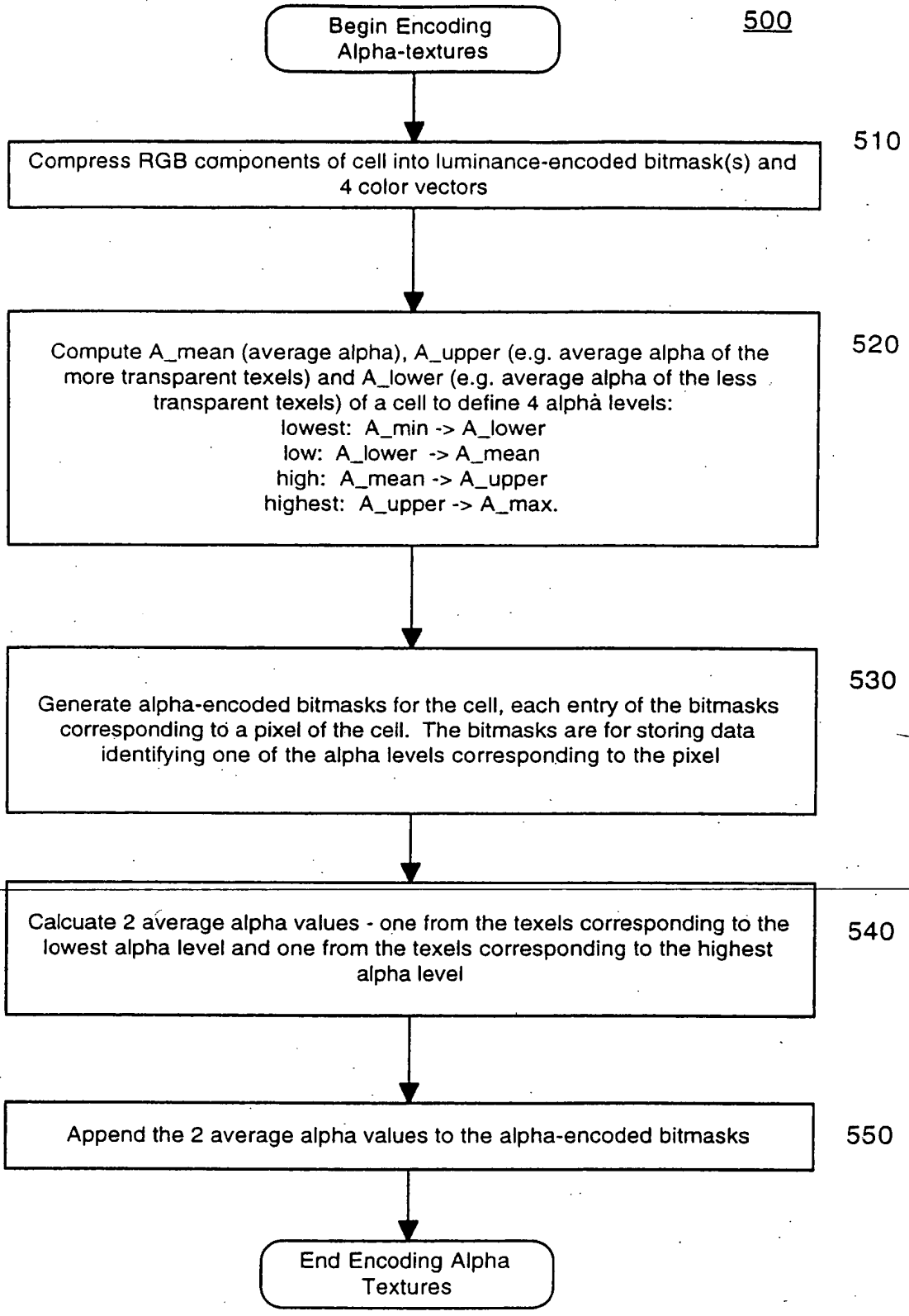
500
Begin Encoding Alpha-textures
510
Compress RGB components of cell into luminance-encoded bitmask(s) and 4 color vectors
520
Compute A_mean (average alpha), A_upper (e.g. average alpha of the more transparent texels) and A_lower (e.g. average alpha of the less transparent texels) of a cell to define 4 alpha levels:
lowest: A_min -> A_lower
low: A_lower -> A_mean
high: A_mean -> A_upper
highest: A_upper -> A_max.
530
Generate alpha-encoded bitmasks for the cell, each entry of the bitmasks corresponding to a pixel of the cell. The bitmasks are for storing data identifying one of the alpha levels corresponding to the pixel
540
Calcuate 2 average alpha values - one from the texels corresponding to the lowest alpha level and one from the texels corresponding to the highest alpha level
550
Append the 2 average alpha values to the alpha-encoded bitmasks
End Encoding Alpha Textures

FIG. 4

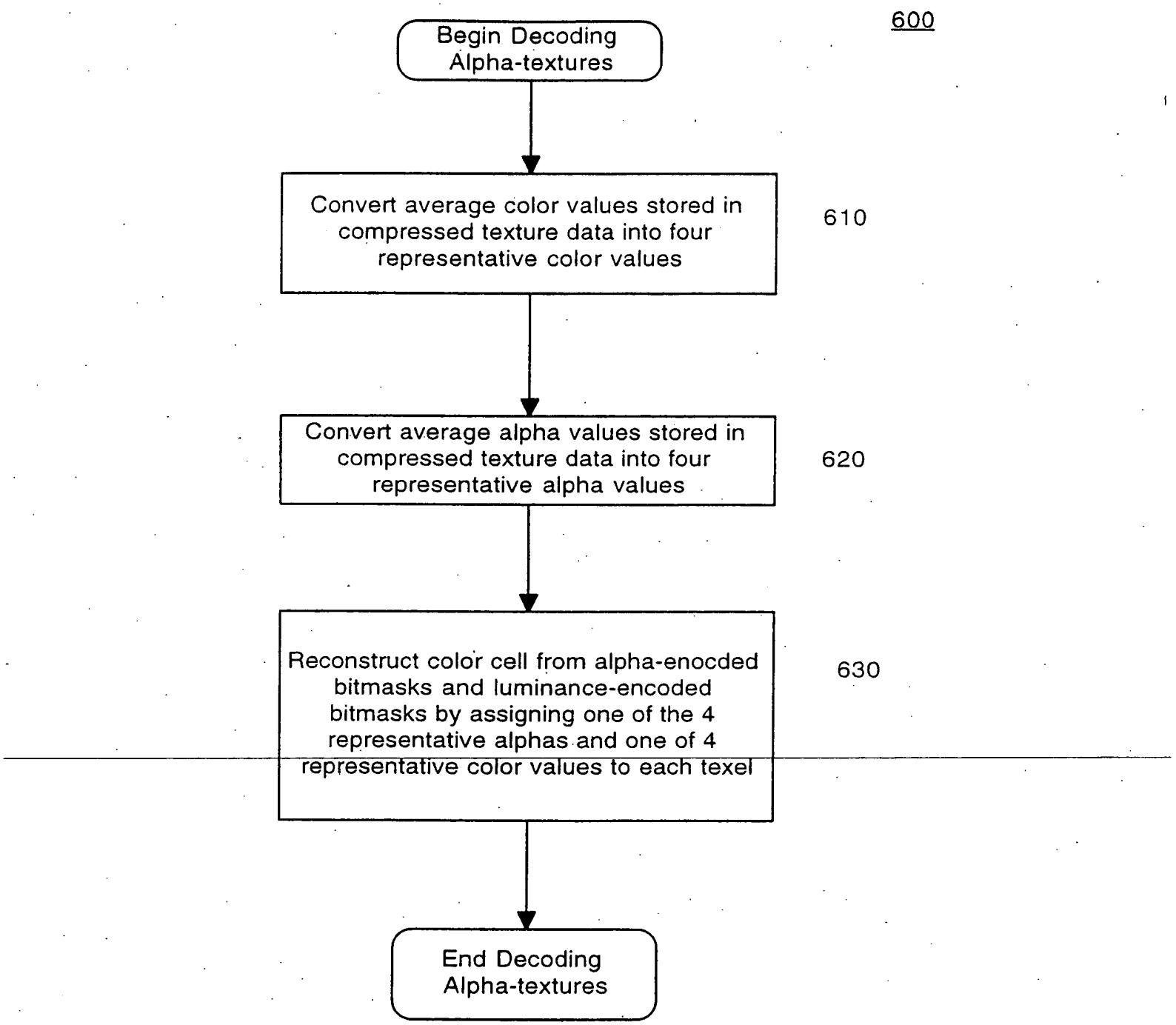
600
Begin Decoding Alpha-textures
Convert average color values stored in compressed texture data into four representative color values
610
Convert average alpha values stored in compressed texture data into four representative alpha values
620
Reconstruct color cell from alpha-enocded bitmasks and luminance-encoded bitmasks by assigning one of the 4 representative alphas and one of 4 representative color values to each texel
630
End Decoding Alpha-textures

FIG. 5

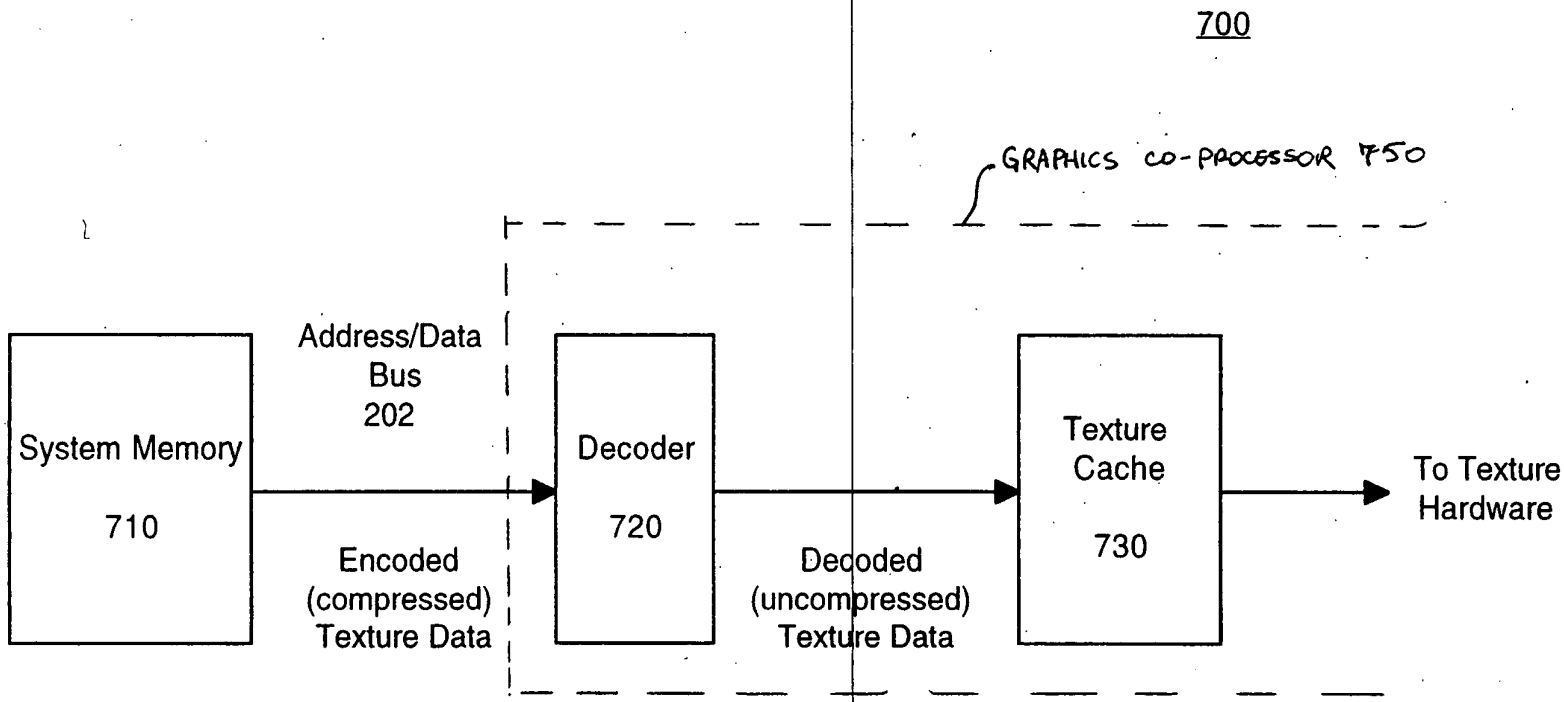
700
GRAPHICS CO-PROCESSOR 750
System Memory
710
Address/Data
Bus
202
Encoded
(compressed)
Texture Data
Decoder
720
Decoded
(uncompressed)
Texture Data
Texture
Cache
730
To Texture
Hardware

FIG. 6

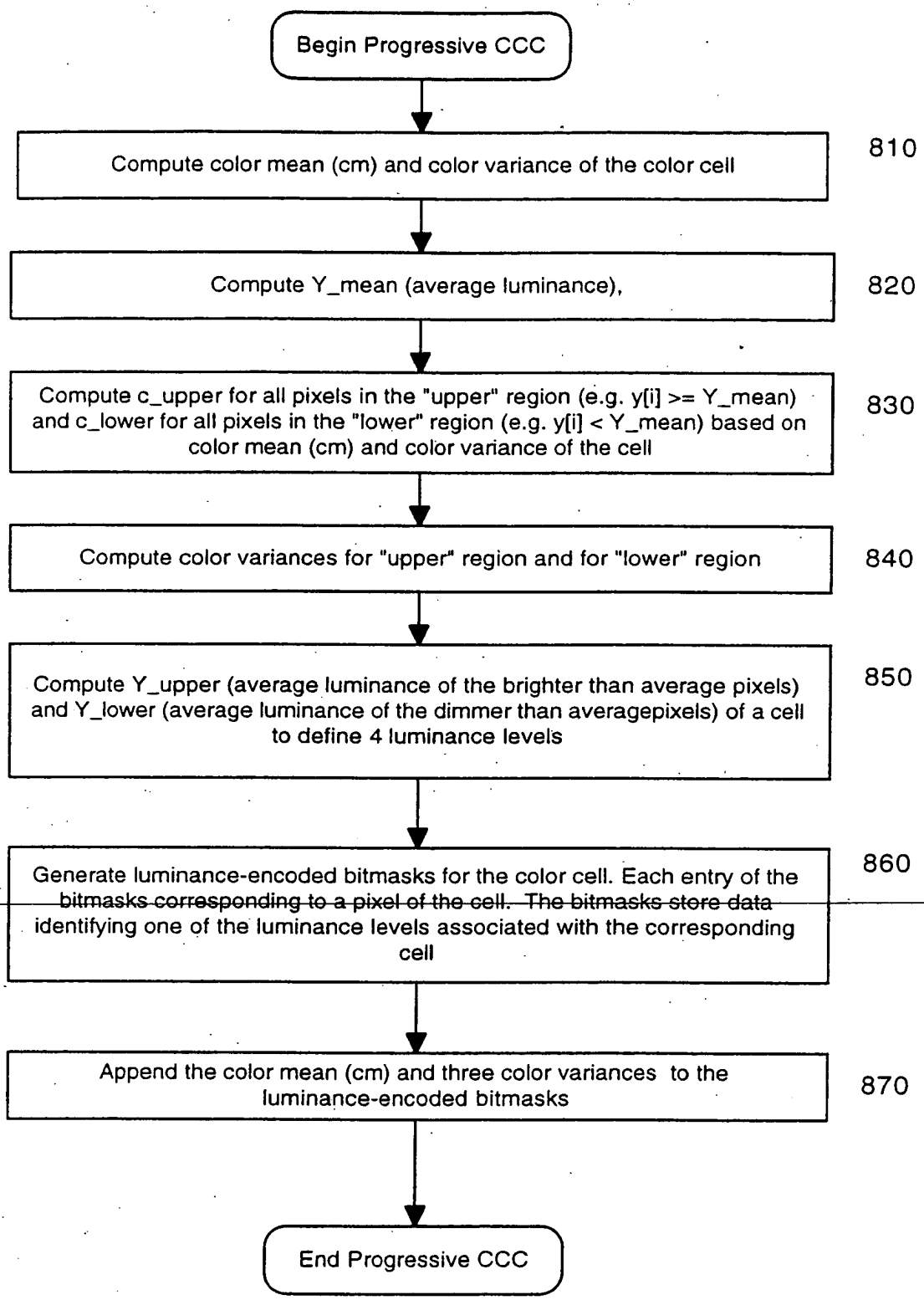
800
Begin Progressive CCC
Compute color mean (cm) and color variance of the color cell
810
Compute Y_mean (average luminance),
820
Compute c_upper for all pixels in the "upper" region (e.g. y[i] >= Y_mean) and c_lower for all pixels in the "lower" region (e.g. y[i] < Y_mean) based on color mean (cm) and color variance of the cell
830
Compute color variances for "upper" region and for "lower" region
840
Compute Y_upper (average luminance of the brighter than average pixels) and Y_lower (average luminance of the dimmer than averagepixels) of a cell to define 4 luminance levels
850
Generate luminance-encoded bitmasks for the color cell. Each entry of the bitmasks corresponding to a pixel of the cell. The bitmasks store data identifying one of the luminance levels associated with the corresponding cell
860
Append the color mean (cm) and three color variances to the luminance-encoded bitmasks
870
End Progressive CCC

FIG. 7

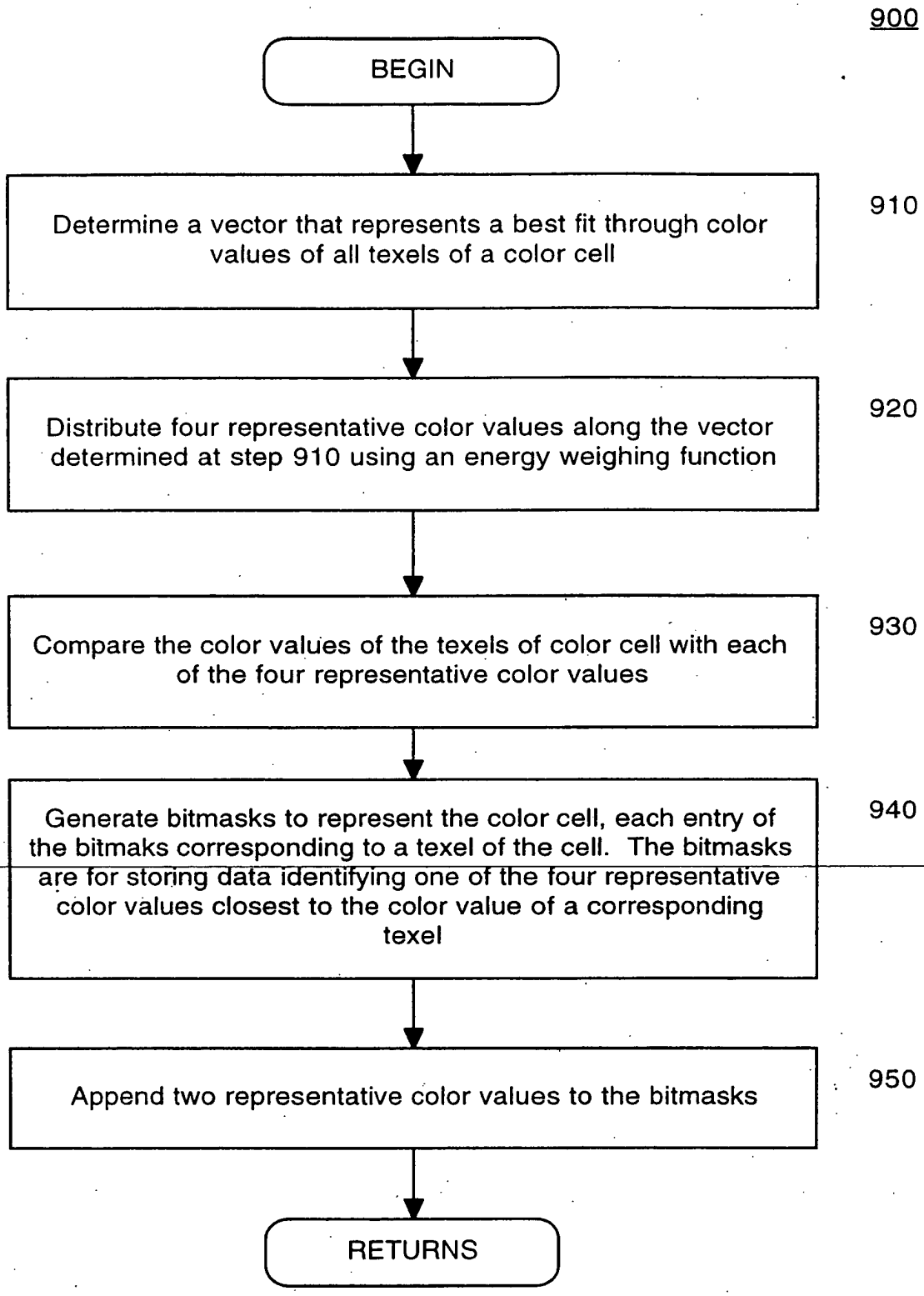
900
BEGIN
Determine a vector that represents a best fit through color values of all texels of a color cell
910
Distribute four representative color values along the vector determined at step 910 using an energy weighing function
920
Compare the color values of the texels of color cell with each of the four representative color values
930
Generate bitmasks to represent the color cell, each entry of the bitmaks corresponding to a texel of the cell. The bitmasks are for storing data identifying one of the four representative color values closest to the color value of a corresponding texel
940
Append two representative color values to the bitmasks
950
RETURNS

FIG. 8